NATIONAL PRACTITIONER DATA BANK (NPDB)
AND
HEALTHCARE INTEGRITY AND PROTECTION DATA BANK (HIPDB)

QUERYING AND REPORTING XML SERVICE (QRXS) CLIENT PROGRAM USER GUIDE

Version 1.03

July 2006

U.S. Department of Health and Human Services
Health Resources & Services Administration
Bureau of Health Professions
Office of Workforce Evaluation and Quality Assurance
Practitioner Data Banks Branch
Parklawn Building, Room 8-103
5600 Fishers Lane
Rockville, Maryland 20857

SUMMARY OF CHANGES – VERSION 1.03

Below is a summary of changes to the Querying and Reporting XML Service (QRXS) User Guide version 1.03. Effective July 31, 2006, this document version 1.03 replaces version 1.02. The changes in this version are indicated below:

- Updated installation instructions for JRE 1.5 and the QRXS client software. See Sections 2.2 and 2.3.

SUMMARY OF CHANGES – VERSION 1.02

Below is a summary of changes to the Querying and Reporting XML Service (QRXS) User Guide version 1.02. Effective May 8, 2006, this document version 1.02 replaces version 1.01. The changes in this version are indicated below:

- The Data Banks' Web site is now located at www.npdb-hipdb.hrsa.gov. The Data Banks are using a .gov domain name to help prevent fraud by showing Data Banks' users that the NPDB-HIPDB Web site is under the Government-run domain. Please update your Internet bookmarks to reference the .gov address for the Data Banks' Web site. NPDB-HIPDB Web site references in this document now refer to the new Web site address.
- Due to the NPDB-HIPDB Web site address change, all ITP and QRXS client programs must be upgraded to a new version. Updated client programs are now available on the NPDB-HIPDB Web site. While the current versions of the ITP and Querying and Reporting XML Service (QRXS) client programs will continue to function for a limited time, all ITP and QRXS users must upgrade their client program to the new version no later than September 18, 2006.
- Updated the commons-fileupload jar to version 1.1.

SUMMARY OF CHANGES – VERSION 1.01

Below is a summary of changes to the Querying and Reporting XML Service (QRXS) User Guide version 1.01. Effective October 17, 2005, this document version 1.01 replaces version 1.0. The changes in this version are indicated below:

- Updated sample code for obtaining error message text using the Application Programming Interface (API). Sections 5.3.3 and 5.3.4.
- Modified and added status codes. See Section 3.3.

Table of Contents

1. Introduction

1.1 Scope

The National Practitioner Data Bank-Healthcare Integrity and Protection Data Bank (NPDB-HIPDB) provides the Querying and Reporting XML Service (QRXS) to users who wish to send and receive machine-readable responses. These instructions provide details on installing and configuring the QRXS client software.

The QRXS client is a stand-alone program that transmits files containing query and report data to, and receives response files from, the Data Banks. For security, data is transmitted over a Secure Socket Layer (SSL) connection. This program can be executed as a stand-alone program, or it can be executed under the control of other programs. An Application Programming Interface (API) is also available for additional flexibility when integrating QRXS transactions into existing software systems.

File formats for QRXS submissions and responses are defined in the Interface Control Documents (ICDs) published by the Data Banks, available on-line at *www.npdb-hipdb.hrsa.gov/qrxs.html.*

1.2 Conventions

Names of files, URLs, parameters, example Data Bank Identification Numbers (DBIDs), example passwords, or documents are presented in italics.

http://java.sun.com/

Data that must be entered into the computer, commands you must enter, contents of files, and contents of directories are presented in a different font (Courier New).

```
java com.npdbhipdb.QRXS send
```

Where the explicit use of upper or lower case letters are necessary due to the system interpreting upper and lower case letters as distinct characters, the text will clearly identify the command as “case sensitive.”

2. Installation

2.1 System Requirements

1. One of the following supported Microsoft Windows versions: Windows 2000, Windows XP. The software can be expected to run on other platforms that support the required Java Runtime Environment (JRE), but only the supported versions have been tested.
2. Access to the Internet and SSL port (443).

2.2 Software Requirements

1. JRE version 1.5.0 or higher.

   The JRE can be downloaded from Sun's JRE download page:

   java.sun.com/javase/downloads

   Follow the instructions provided on the web page to download and install the JRE. If the default installation is followed, the JRE will install to a directory specific to the exact release number. For example, if the release is at version 1.5.0_06, the default installation will create a directory named *C:\Program Files\Java\1.5.0_06*.

2. The *QRXS-v1_03.zip* distribution available at *www.npdb-hipdb.hrsa.gov/qrxs.html.*

   The following files are provided in the zip file:

   commons-fileupload-1.1.jar

   commons-httpclient-2.0.jar

   commons-logging.jar

   QRXS-Javadoc-Files-v1_02.zip

   QRXS-Readme.txt

   qrxs-v1_03.jar

   qrxs.properties

2.3 Software Installation

1. Install the JRE. Instructions are provided at Sun's JRE download page, referenced above.

2. Create a working directory for running the client. For example:

   ```
   mkdir c:\QRXS
   ```

3. Install the JAR files from the *QRXS-v1_03.zip* distribution. Copy the .jar files (*qrxs-v1_03.jar, commons-fileupload-1.1.jar, commons-httpclient-2.0.jar,* and *commons-logging.jar)* to the JRE's library extension directory. For example (using the default JRE installation directory):

   c:\Program Files\Java\1.5.0_06\lib\ext

4. Install the initialization file. Copy the file *qrxs.properties* to the directory from which the QRXS client executes (your working directory, e.g., *c:\QRXS*).

Note: Previous versions of QRXS-related files should be removed before installing newer versions in order to ensure the latest files are used.

3. Client Configuration and Execution

3.1 Executing the QRXS Client

The client is controlled via command-line parameters. The following are the only supported commands:

`java com.npdbhipdb.QRXS send` (for sending submission files) or
`java com.npdbhipdb.QRXS receive` (for receiving response files) or
`java com.npdbhipdb.QRXS encode <password>` (for encoding a password in the initialization file)

Note: The program name and the command-line parameter values are case sensitive.

3.2 Preparing the Initialization File

On startup, the *qrxs.properties* file is read for initialization information. The *qrxs.properties* file resides in the working directory as described in Section 2.3, Software Installation, Step 4. A sample *qrxs.properties* file is provided in the zip file. All the required properties in the initialization file must be complete and valid. If the information is missing or invalid, the program terminates. Users may configure their own *qrxs.properties* files to reflect their desired download directory location, upload/download list file locations, and to specify use of a proxy server.

Certain characters may be interpreted as instructions by the user's operating system or by the QRXS client. For this reason, the *qrxs.properties* file must be written in such a way that each character is interpreted literally, and not as an instruction. In order to accomplish this, characters that may be interpreted as instructions must be escaped; i.e., a backslash ("\") character is placed in front of the character that may be interpreted as an instruction.

The following is a sample *qrxs.properties* file:

```
DBID=123400000000123
Password=UuhUh5owKFEeue
UserID=myuserid
DownloadDir=c:\\QRXS\\download
UploadListFile=upload.lst
DownloadListFile=download.lst
```

Notes:

1. The properties are case sensitive.

2. The password is encoded and **must be set using the client's encode command**. Instructions on setting the password are in Section 3.7, Encoding the Password.

3. The backslash character itself must be escaped, because in this case it must be interpreted literally; this is the reason for the two backslashes in the directory values.

4. The properties within the file may appear in any order.

The following properties are allowed. Users must enter any properties that they do not wish to use the default values for:

Properties	Descriptions
DBID*	The Data Bank Identification Number (DBID).
UserID*	The User ID of a user registered with the Data Banks. Do not use the Administrator's ID.**
Password*	The encoded Data Bank password (also known as the IQRS password).
DownloadDir*	Directory to hold the downloaded QRXS response files .
UploadListFile	Name of the file containing one or more names of QRXS submission files; defaults to *upload.lst.*
DownloadListFile	File containing names of the QRXS response files downloaded; defaults to *download.lst.*
LogFile	Name of the file containing error messages; defaults to *qrxs.log.*
ProxyHostName	Name of the proxy host (usually an IP Address). Only required when using a proxy server (see Section 4.4, Proxy Server for more information).
ProxyPort	Port number of the proxy server. Only required when using a proxy server (see Section 4.4, Proxy Server for more information).
* Required property. ** For detailed information on maintaining User IDs and passwords refer to *www.npdb-hipdb.hrsa.gov/iqrs.html.* For *UploadListFile, DownloadDir, LogFile,* and *DownloadListFile*, acceptable values of the properties include the file name only, or a path name and the file name. If the value is the file name only, then the file is assumed to be located in the working directory.	

3.3 Status Codes

The QRXS client provides status in three ways. First, an exit code containing the status code value is always returned; second, a file called *qrxsstat.dat*, which contains the status code, is created in the program's working directory; and third, non-zero status codes and corresponding messages are appended to a log file located in the program's working directory. A status code of "0" indicates that the transaction was successful, and a negative number status code indicates that the transaction failed.

The following status codes are returned:

Status Code	Description
0	Success, no errors.
-1	Error on NPDB-HIPDB server, reattempt transfer.
-2	Authentication failed; invalid DBID, UserID, or Password.*
-3	Password expired for DBID and UserID.*
-4	No filenames specified for upload.
-5	Error sending files to server, reattempt transfer.
-6	Client sent an invalid request.
-7	Inactive DBID.
-8	Maximum upload file size exceeded; reduce file size and reattempt transfer.
-9	All uploaded files are invalid.
-10	Unable to communicate with NPDB-HIPDB server, reattempt transfer.
-12	Database error on NPDB-HIPDB server, reattempt transfer.
-13	Not all of the files were processed successfully.

Status Code	Description
-14	Due to the publication of final regulations implementing Section 1921 of the *Social Security Act*, the DBID for your organization must be renewed before the submission file can be processed by the Data Banks. The certifying official for your organization must review the new Section 1921 statutory authorities, available at http://www.npdb-hipdb.hrsa.gov/legislation.html, as part of the renewal process. Once the statutory authorities have been reviewed, complete the on-line registration renewal form by logging in to the IQRS and selecting Renew Registration on the registration confirmation screen. The completed form must be signed and mailed to the NPDB-HIPDB for processing. If your organization has already mailed your registration renewal to the Data Banks, it will be processed within one business day of its receipt by the NPDB-HIPDB. You will receive Data Bank Correspondence once the Data Banks have successfully processed your registration renewal form. If necessary, you may complete a new form by selecting Renew Registration on the registration confirmation screen within the IQRS. If you need further assistance, please contact the NPDB-HIPDB Customer Service Center at 1-800-767-6732.
-15	Due to the publication of final regulations implementing Section 1921 of the *Social Security Act*, the DBID for your organization must be renewed before the submission file can be processed by the Data Banks. The certifying official for your organization must review the new Section 1921 statutory authorities, available at http://www.npdb-hipdb.hrsa.gov/legislation.html, as part of the renewal process. Contact the administrator of your organization so they can renew the registration. If you need further assistance, please contact the NPDB-HIPDB Customer Service Center at 1-800-767-6732.
-16	The DBID for your organization must be renewed before you can access the Data Banks' services. The NPDB-HIPDB requires all registered entities to periodically renew their registration information. Re-registration enables the NPDB-HIPDB to maintain accurate entity contact information and provides the entity with the opportunity to review the legal requirements and verify their compliance for participation with NPDB-HIPDB. The certifying official for your organization must review the NPDB-HIPDB statutory authorities, available at http://www.npdb-hipdb.hrsa.gov/legislation.html, as part of the renewal process. Once the statutory authorities have been reviewed, complete the on-line registration renewal form by logging in to the IQRS and selecting Renew Registration on the registration confirmation screen. The completed form must be signed and mailed to the NPDB-HIPDB for processing. If your organization has already mailed the registration renewal to the Data Banks, it will be processed within one business day of its receipt by the NPDB-HIPDB. Data Bank Correspondence will be sent once the Data Banks have successfully processed your registration renewal form. If necessary, you may complete a new form by selecting Renew Registration below. If you need further assistance, please contact the NPDB-HIPDB Customer Service Center at 1-800-767-6732.
-17	The DBID for your organization must be renewed before you can access the Data Banks' services. The NPDB-HIPDB requires all registered entities to periodically renew their registration information. Re-registration enables the NPDB-HIPDB to maintain accurate entity contact information and provides the entity with the opportunity to review the legal requirements and verify their compliance for participation with NPDB-HIPDB. The certifying official for your organization must review the NPDB-HIPDB statutory authorities, available at http://www.npdb-hipdb.hrsa.gov/legislation.html, as part of the renewal process. Contact the administrator of your organization so they can renew the registration. If you need further assistance, please contact the NPDB-HIPDB Customer Service Center at 1-800-767-6732.
-21	Client error, refer to log file for more information.
-22	Usage error, check command-line parameter.
-23	Error in reading Initialization file *qrxs.properties*, check that *qrxs.properties* file is in working directory.
-24	Unable to open program log file, check that disk space is available.
-30	Error validating property in *qrxs.properties*, a property is missing or value is invalid.
-31	Error validating *UploadListFile*, upload file missing or a file listed in the upload file does not exist.
-32	Error validating *DownloadDir*, directory does not exist.
-40	Error opening connection to NPDB-HIPDB server, check Internet connection and reattempt transfer.
-43	Error during client startup.**
-54	Error downloading response files, reattempt transfer.

Status Code	Description
-56	Error getting the list of response files downloaded, reattempt transfer.
* For detailed information on maintaining User IDs and passwords refer to *www.npdb-hipdb.hrsa.gov/iqrs.html.* ** Either two instances of the client program are running or one instance of the program was terminated prematurely. If it is the latter, remove the qrxslock file from the program's working directory and try again.	

3.4 QRXS Status File

The *qrxsstat.dat* file contains the final status of the QRXS client execution. This file is created in the working directory immediately upon program completion. The file contains two lines. The first line indicates the overall status of "Done" or "Not Done." The second line contains a status code of "0" or a negative number (see Section 3.3, Status Codes, for error codes that may be returned).

Sample *qrxsstat.dat* file for a successful execution:

```
Overall status        = Done
StatusCode=0
```

Sample *qrxsstat.dat* file for an unsuccessful execution:

```
Overall status        = Not Done
StatusCode=-22
```

3.5 Uploading Files

The upload file contains the names (including the full or relative path) of the submission files that are sent to the Data Bank(s). The name of the upload file must be specified in the *qrxs.properties* file using the *UploadListFile* property. If it is not specified, it defaults to *upload.lst.*

A sample upload file with four submission files to upload, using a full path:

```
c:\QRXS\upload\0000001.xml
c:\QRXS\upload\0000002.xml
c:\QRXS\upload\0000004.xml
c:\QRXS\upload\0000005.xml
```

A sample upload file with three submission files to upload, using a relative path:

```
upload\file0001.xml
upload\file0002.xml
upload\file0003.xml
```

Notes:

1. In the upload file, the backslash is not escaped.

2. A file name can contain a relative or full path as noted above. Each set of files above is specified differently, but all the files are located in *c:\QRXS\upload*, and the working directory is *c:\QRXS*.

3. Submission files must not exceed 3 MB.

3.6 Downloading Files

Response files are stored in the download directory after being successfully received from the Data Bank(s). The names of the files are in the following format: ####################*.xml* (i.e., a unique 20-digit ID number with the '*xml'* extension).

The download list file contains the names of downloaded files. The name of the download list file must be specified in the *qrxs.properties* file with the *DownloadListFile* property. If it is not specified, it defaults to *download.lst.*

The following is the content of a download list file that might be created:

```
c:\QRXS\download\10000000000000000001.xml
c:\QRXS\download\10000000000000000002.xml
c:\QRXS\download\10000000000000000003.xml
c:\QRXS\download\10000000000000000004.xml
```

Notes:

1. The downloaded response files have unique names. If files from a previous download exist in the directory, then the new files will be added to the directory without affecting the existing files.

2. The download list file is recreated on every successful download of response files. The download list file is not created if there are no response files to download or if there was an error downloading response files.

3. Response files will not exceed 1MB. Responses that are larger than 1MB are separated into multiple response files.

4. Responses are available for download within an average of two to four hours after submission. Under certain circumstances, additional processing time may be required.

3.7 Encoding the Password

In order to maintain the integrity and confidentiality of the Data Bank information, the QRXS initialization file must contain a valid encrypted Data Bank password. A valid password and UserID is required for the successful execution of all upload and download transactions. The password is associated with a registered Data Bank user, and will automatically expire every 90 days. The command listed below must be executed prior to the first upload or download transaction, and each time the Data Bank user updates their password (a minimum of every 90 days).

The password is encoded by executing the encode command of the client and providing the password as a command-line parameter. The password is encoded and written to *qrxs.properties*. **The password must be entered using this encode method for login validation to succeed.** The following is an example of how to perform the encoding:

```
java com.npdbhipdb.QRXS encode myPassw0rd
```

The encrypted password is then written to the *qrxs.properties* file:

```
Password= 2MhEB2Fsue1I7eqJLDyI
```

Notes:

1. The command and the password are case sensitive.

2. The encoding process may cause some characters in *qrxs.properties* to be escaped; this is denoted by a backslash in front of the escaped characters.

3. The encoding process may cause the lines in *qrxs.properties* to be reordered; this does not affect the operation of the client.

4. Sample Configuration and Execution

The following sections guide you through a sample configuration and execution of the QRXS client. This is only a sample configuration and may be changed if desired.

4.1 Configuration

The client is installed in the directory *c:\QRXS* for entity DBID *123400000000123, UserID BobJones,* and password *myPassw0rd.* The initialization file is configured with the download directory *c:\QRXS\download*, upload file *updlist.fil*, download list file *qrxsresp.fil*, and log file *qrxserrors.fil.* The submission files for upload are located in *c:\QRXS\upload*. The user creates the new directories:

```
cd QRXS

mkdir download

mkdir upload
```

The following is a listing of files in directory *c:\QRXS*:

```
qrxs.properties
<DIR>  download
<DIR>  upload
```

The working directory is *c:\QRXS*. Encode the password by executing the following command:

```
java com.npdbhipdb.QRXS encode myPassw0rd
```

The following is a printout of the *qrxs.properties* initialization file (the order of properties may vary; this is normal):

```
UploadListFile=updlist.fil
Password= 2MhEB2Fsue1I7eqJLDyI
DownloadDir=download
DBID=123400000000123
UserID=BobJones
DownloadListFile=qrxsresp.fil
LogFile=qrxserrors.fil
```

4.2 Uploading Submission Files

The submission files must first be uploaded to the Data Bank(s). Submission files *0000001.xml, 0000002.xml*, *0000003.xml*, and *0000004.xml* are placed in the upload directory *c:\QRXS\upload*. The upload file, *updlist.fil*, is placed in the working directory *c:\QRXS*. The following is a printout of *updlist.fil*:

```
c:\QRXS\upload\0000001.xml
c:\QRXS\upload\0000002.xml
c:\QRXS\upload\0000003.xml
c:\QRXS\upload\0000004.xml
```

Execute the following command, either from the command prompt or from a user application, to upload files to the Data Bank(s). This command is case sensitive and must be executed exactly as shown here:

```
java com.npdbhipdb.QRXS send
```

After execution of this command, the submission files are uploaded to the Data Bank(s) successfully. **The submission may generate some immediate responses**, depending on the processing required to handle the submissions. The number of responses received will be displayed:

```
QRXS Sending…
Received 2 response(s)
Status Code: 0
```

The response files are downloaded to the download directory, *c:\QRXS\download*. The download list file *qrxsresp.fil* is created in the directory *c:\QRXS*.

The file *qrxsstat.dat* is created in the working directory *c:\QRXS*. The *qrxsstat.dat* file indicates a successful status code of "0." The following is a printout of the *qrxsstat.dat* file that is created:

```
Overall status        = Done
StatusCode=0
```

4.3 Downloading Response Files

The remaining response files may be downloaded after they have completed processing. To download response files, execute the QRXS client in receive mode:

```
java com.npdbhipdb.QRXS receive
```

After execution of this command the response files are downloaded from the Data Bank(s) successfully. The number of responses received will be displayed:

```
QRXS Receiving...
Received 4 response(s)
Status Code: 0
```

The four response files are downloaded to the download directory, *c:\QRXS\download*. The directory *c:\QRXS\download* contains the following files:

```
10000000000000000001.xml
10000000000000000002.xml
10000000000000000003.xml
10000000000000000004.xml
```

The download list file *qrxsresp.fil* is created in the directory *c:\QRXS*. The following is a printout of the *qrxsresp.fil* file:

```
c:\QRXS\download\10000000000000000001.xml
c:\QRXS\download\10000000000000000002.xml
c:\QRXS\download\10000000000000000003.xml
c:\QRXS\download\10000000000000000004.xml
```

The status file *qrxsstat.dat* is also created in the directory *c:\QRXS*. The following is a printout of the *qrxsstat.dat* file that is created:

```
Overall status        = Done
StatusCode=0
```

4.4 Proxy Server

A proxy server is a program that mediates requests between the local network and the workstations on them and the Internet beyond. Contact your network administrator for proxy server information. To enable the QRXS to work with proxy servers, the *qrxs.properties* file needs the parameters *ProxyHostName* (specifies the host name of the proxy server; usually an IP Address), and *ProxyPort* (specifies the port number the proxy server connects to). Once these parameters are set, sending and receiving from the QRXS proceeds as normal.

The following is a printout of the *qrxs.properties* initialization file for use with a Proxy Server:

```
UploadListFile=updlist.fil
Password= 2MhEB2Fsue1I7eqJLDyI
DownloadDir=download
DBID=123400000000123
UserID=BobJones
DownloadListFile=qrxsresp.fil
LogFile=qrxserrors.fil
ProxyHostName=111.111.111.111
ProxyPort=1000
```

5. Application Programming Interface

5.1 Overview

The QRXS Application Programming Interface (API) is a Java-based library that provides the capability to interface with the QRXS. The API provides the same capabilities available in the QRXS client program. In fact, the QRXS client is built on top of the API. Javadocs for the QRXS API can be found on-line at *www.npdb-hipdb.hrsa.gov/qrxs.html*.

5.2 When to use the API

The provided client program is appropriate for most users. The API may be beneficial for users who wish to integrate their QRXS interaction into existing Java software. Submitting files and examining responses and error codes can be done programmatically, eliminating the need to write code to parse the text results that are generated by the client program.

5.3 Using the API

Install the libraries as described in Section 2.3, Software Installation. The QRXS API is in the `com.npdbhipdb` package. Import the package to make the API visible to the Java application.

```
import com.npdbhipdb.*;
```

5.3.1 Configuration

The QRXS Configuration class contains the properties needed to communicate with the QRXS. An instance of the Configuration is created and the object is passed to the send() or receive() method. A Configuration instance can be created using an existing initialization file (Section 3.2, Preparing the Initialization File), or by providing the property values directly during instantiation.

5.3.1.1 Using a File

A Configuration object can be created using the same properties file format (Section 3.2, Preparing the Initialization File) used for the client application. The **password must be encoded** in the file using the client's encode command (Section 3.7, Encoding the Password) or the API encode() method (Section 5.3.2, Encoding). Exceptions can be thrown if a required property is not defined or if the file cannot be opened.

```
final String configFilename = "qrxs.properties";

// create an instance of QRXS
QRXS qrxsClient = new QRXS();
QRXS.Configuration config;

try
{
    // create Configuration using the properties file
    config = qrxsClient.new Configuration(configFilename);
}
catch (QRXS.Configuration.InvalidConfigurationException e)
{
    // configuration property missing
    System.out.println("Error: " + e.getCode());
}
catch (QRXSException e)
{
    // problem encountered opening the file
    System.out.println("Error: " + e.getCode());
}
```

5.3.1.2 Using Parameters

A Configuration object can be created by specifying the required properties as parameters to the constructor. With this method there is no need to perform the password encoding required when using a file. An exception can be thrown if a property value is invalid.

```
final String entityDBID = "123456787654321";
final String userID  = "myusername";
final String password   = "myPassw0rd";
final String downloadDir = "C:/QRXS/download";
QRXS.Configuration config;

// create an instance of QRXS
QRXS qrxsClient = new QRXS();

try
{
    // create Configuration using the required values
    config = qrxsClient.new Configuration(entityDBID, userID, password,
                                          downloadDir);
}
catch (QRXS.Configuration.InvalidConfigurationException e)
{
    // configuration property invalid
    System.out.println("Error: " + e.getCode());
}
```

5.3.2 Encoding

Encoding is used to write an encrypted password to a configuration properties file. The file will be re-written with the password value encoded. The ordering of the properties in the file may be changed by the encoding process. The password in a configuration file **must be encoded** before sending and receiving or authentication will fail.

```
final String password   = "myPassw0rd";
final String configFilename = "qrxs.properties";

// create an instance of QRXS client
QRXS qrxsClient = new QRXS();

try
{
    qrxsClient.encode(configFilename, password);
}
catch (QRXSException e)
{
    // unable to read/write the configuration file
    System.out.println("Error: " + e.getCode());
}
```

5.3.3 Sending

The send process involves establishing the connection to the QRXS, uploading the specified submission files, and receiving responses.

```
// create Lists to hold submissions and responses
List responseFilenames = new ArrayList();
List submissionFilenames = new ArrayList();
List errorFiles = new ArrayList();
int status;

// populate the list of file(s) to upload
submissionFilenames.add("C:/QRXS/upload/000000001.xml");
// add the remaining files
try
{
    // send the files
    status = qrxsClient.send(config, submissionFilenames,
                             responseFilenames, errorFiles);
}
catch (QRXSException e)
{
    System.out.println("Error Code: " + e.getCode());
    System.out.println("Error Message: " + e.getMessage());
}
```

Files returned in the response are stored to the location specified in the Configuration object. The `responseFilenames` List contains the names of the response files. Sent files that could not be processed are listed with the accompanying error code in a list of `ErrorFile` objects. Each `ErrorFile` object also contains one or more error messages giving a textual description of the error.

```
Iterator fileIter = errorFiles.iterator();

while (fileIter.hasNext())
{
    QRXS.ErrorFile err = (QRXS.ErrorFile) fileIter.next();
    System.out.println("File " + err.getFilename() + ", " + err.getCode());

    String errorMessages[] = err.getErrorMessages();
    for (int i=0; i<errorMessages.length; i++)
    {
        System.out.println("Error Message: " +
                           MessageFormat.format(errorMessages[i], new String[]
                           {err.getFilename()}));
    }
}
```

5.3.4 Receiving

The receive process establishes the connection to the QRXS and downloads any response files that have completed processing by the Data Banks.

```
// create List to hold responses
List responseFilenames = new ArrayList();
int status;

try
{
    status = qrxsClient.receive(config, responseFilenames);
}
catch (QRXSException e)
{
    System.out.println("Error Code: " + e.getCode());
    System.out.println("Error Message: " + e.getMessage());
}
```

Files returned in the response are stored to the location specified in the Configuration object. The `responseFilenames` List contains the names of the response files.